ONE MODEL TO RULE THEM ALL: UNIFIED CLASSIFICATION MODEL FOR GEOTAGGING WEBSITES

Alexey Volkov
e-mail: ark-kum@yandex-team.ru

Pavel Serdyukov
e-mail: pavser@yandex-team.ru

Michael Maslov
e-mail: maslov@yandex-team.ru

Yandex LLC
Russian Federation

Abstract

The paper presents a novel approach to finding regional scopes (geotagging) of websites. It relies on a single binary classification model per region type to perform the multi-label classification and uses a variety of different features that have not been yet used together for machine-learning based regional classification of websites. The evaluation demonstrates the advantage of our *one model per region type* method versus the traditional *one model per region* approach.

Keywords: *geotagging, classification models, entities, machine learning*

1. INTRODUCTION

One of the problems that search systems face nowadays is the problem of search localization. Users enter search queries that need different answers depending on the location of the user. The relevant result for the query "president's website» depends on the user's country, and the «pizza delivery» query is even more location-specific. Many web resources (websites of local businesses, restaurants or theaters) are relevant only to the users from some specific location. To provide relevant results for the region-specific queries, the search engine needs to know not only the user's location, but also the regional focus of the websites.

The problem we address in this paper is the detection of the regional focus (geotagging) of a website. For each website we want to find the set of regions where the people, interested in the website's content, are located. The solution to the problem of finding a geographical focus of a web page was first proposed by Ding et al. [4]. Their approach was largely based on disambiguation of toponyms or highly location specific entities (e.g. phones and zip codes) in the web-page's content. The follow-on work by Amitay et

al. [1] and Zong et al. [10] relied on propagating the confidence weights of detected toponyms up to the root of the gazetteer taxonomy to find the most probable common ascendants (e.g. finding a country for several cities mentioned). The work by Pyaling et al. [12] combined different features to sequentially refine the result. All these works did not consider any machine-learning based approach to regional classification and ignored any terms appearing on a page that were not known to be location-specific entities in advance. However, a number of approaches to regional classification of other types of web content, for example, user-tagged online photos [3][9] or tweets [2], followed a more traditional approach to text classification and learned a classification function for each region present in the training set of geotagged resources [7], using any terms as features. While being more theoretically grounded, such an approach obviously suffers from the lack of training data for less popular regions, where people do not provide enough geotagged content to learn a good predictor.

In contrast to the above-mentioned approaches, we propose a classification framework, where both entity- and term-based features are used together to provide a high-quality regional classification of web-sites. Moreover, we rely on a single model per region type (i.e. "cities" or "countries"), rather than on an individual model for each region, which greatly solves the problem of data scarcity and allows for acceptable classification performance even for websites from the regions with just a few or even no geotagged websites in the training data. Our method can be used for regions of any type (country, state, area, city etc.), but the system described in this paper focuses on the two region types – country and city. In this paper the term *region* is general and denotes a city or country.

The paper is organized as follows. The next section contains the description of our classification approach, including the details of the features used and the classification algorithm using these features. Section 3 demonstrates the advantage of our classifier that uses one model per region type over a classifier that builds a model for every region. Section 4 concludes the paper and outlines possible extensions of the presented approach that are left for future work.

2. METHOD

Our goal is to associate a correct set of regions $r(w) \subset R$ with every website $w \in W$, where W is the set of websites and R is a set of regions. This is a standard multi-label classification problem where each object (website) can belong to multiple classes (regions). In this paper we present a system that is able to use machine learning to train a single model and use it to detect

multiple regions with the same or better quality than the traditional "one model per region» system. Instead of training a set of models that classify websites as relevant or irrelevant to each region, our system trains a model that classifies the <site, region> pairs. We train a ranking model which aims to infer the relevance of a region to a website. By applying a classification function, we calculate the probabilistic score for each pair. We would now describe the classification system in more details.

2.1. Classification method

Most of the previous works on regional classification of web resources (see Section 1), that employed machine learning, used a set of binary classifiers with the one-versus-all approach for the task of multi-label classification. There are several problems with this approach. First, such systems can only properly classify websites belonging to the regions present in the training set. Moreover, even if the region is present in the training set, but the number of training examples for the region is small, the classification quality for the website from that region can be poor. Besides, the computational complexity of the "model per region» approach depends linearly on the number of classes and hence on the breadth and the level of granularity of the region taxonomy. The main advantage of our approach to the regional classification problem is using a single model per region type for multi-label classification. More importantly, the models turn out better trained since they have much more positive examples than the "per region» models.

2.1.1. Why a single model?

We want to provide a list of thoughts and considerations that led us to the idea of the "one model per region type» system and relative regional features. Although we mention cities, the same applies to countries or regions of other type.

- City's relation to any other city is generally weak. Thus, to make decision about a city, we do not need the information about any other specific city.
- To detect a city, the information concerning only that city is not enough. We still need information about other cities, but this information can be aggregated.
- Different cities have a lot in common. They are the same type of entity, the same type of region. The models, that detect different cities, should be similar.
- Having a single model to train allows us to train it better. We can utilize all the data from the training dataset, not just the data concerning a single city.

• If the model is generic and not specific to a single city, then the features need to be generic too. We need to construct generic features that have similar values and behavior for different cities.

• How can a single model output what cities it has detected for the website? What should the model output if it detects multiple cities? How can the model tell one city from another if the features behave the same for different cities?

2.1.2. Single model solution

The listed questions and considerations are addressed if we change the object of classification. Instead of training a set of models that classify websites as relevant or irrelevant to the corresponding regions, our system trains a model that classifies the <site,region> pairs. In other words, we train a ranking model which aims to infer the relevance of a region to a website, in a similar way that is followed, for example, by Learning to Rank algorithms [6] to infer the relevance of a website to an arbitrary query. Consequently the multi-class classification problem becomes a binary classification problem, greatly reducing the computational complexity.

2.1.3. A system of classifiers

Countries and cities differ too much as regions, so, while we could use a single model for both countries and cities, it's better to have separate models (e.g. "countries», "cities») for each of these types of regions. Moreover, the problem of determining whether the website is relevant to any specific country (city) or has no regional focus (global or national) is different from the problem of associating a specific country (city) with the website. Therefore we use additional binary classifiers for that task. The purpose of a "national» classifier is to separate the websites that should be associated with some city from the websites that should not be associated with any specific city. The same applies to the "global» classifier. The results of these classifiers are used by other classifiers as single value features. The dependency scheme (as well as the used features) is shown on Figure 1.

2.2. Data sources and features

The feature vectors used in our classification system combine the data external to the website's content with the information, extracted from the full text of the indexed web documents, and the geographic information, mined from the documents using entity extraction techniques. Our system does not use textual features directly, but rather uses the results of a third-party text-based classifier [13] which uses terms as features. The idea was not only to use the

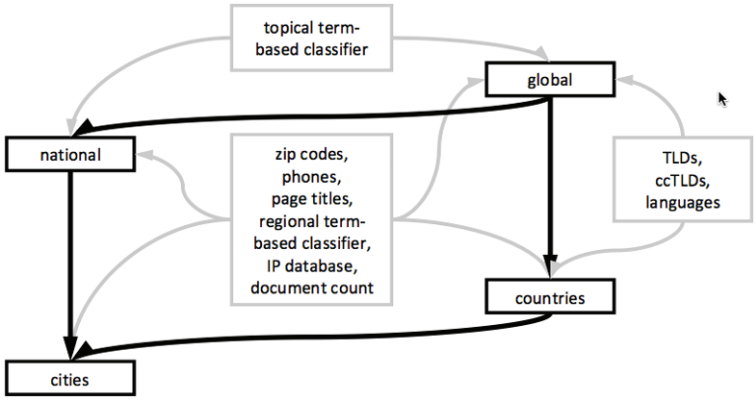

Figure 1. Dependencies between the classification models and features

regional aspect of the textual classification. We wanted to also capture the topical aspects, encoded in the document's words, which can influence the decision of assigning a website to the global or local scope class. For instance, websites about programming are unlikely to be local as opposed to the websites of restaurants or movie theaters. We briefly describe the set of features used in this work. Note that all features, described in this section, appeared to be useful in our preliminary experiments (so, their removal lead to the classification performance degradation). However, we do not provide a detailed analysis of individual feature performance due to space constraints, and as long as such a study is not the primary focus of this paper.

2.2.1. Spectrum features

As the object of classification is the <site, region> pair, we use a variety of features that reflect the degree of match between the website and the region. Further in this section, we use the notion of a *spectrum* to describe a discrete map from a set of classes to a numerical value: *Spectrum*: $C \to \mathbb{R}$, where C is some set of classes (i.e. regions, languages, topics) and $\mathbb{R}$ is the set of real numbers. A *spectrum feature* associates a spectrum with each website. Spectrum feature: $W \to C \to \mathbb{R}$, where W is a set of all websites. One example of spectrum feature is a *number of times a zip code of some region was found on the website*, which gives us a spectrum for each website: *website* → *region* → *number of zip codes*. If the set of classes C is a set of regions, we call such spectrum *regional*. Each spectrum can be viewed on different region scale levels (countries, cities). When a regional spectrum is viewed on the country level, the values for the regions are converted to the corresponding country values and summed.

2.2.2. Geo-coding

Some features have only implicit connection to regions. These features must first be geo-coded. Geo-coding is the process of converting an implicitly regional feature to explicit region-specific factors. One example of regional coding would be converting the feature "Pages of the website mention the "08002 zip code 6 times» to the region-specific feature "Pages of the website mention the zip code of Barcelona 6 times». Regional coding is not always trivial - take the page languages for example. While languages are most certainly related to countries, we cannot just convert the feature "The website has 60 pages in Spanish» to the region-specific feature "The website has 60 pages in the language spoken in Spain», because Spain is not the only country where people use Spanish. Such situations need to be thought over and solved carefully. Regional coding is vital to our system, because the system has to be aware of the regions and their relation. The system needs to know that the features "zip-code 08002 mentioned 6 times», "name Barcelona mentioned 8 times» and "phone code +34(93) mentioned 3 times» all refer to the same region while the feature "zip-code 12345 mentioned 2 times» refers to a different region.

2.2.3. Relative features

On order to create a single model that works for multiple classes, the regional features must be converted from "absolute" features, which mention specific regions, to "relative" features. Formulas for the relative features do not mention specific regions. Instead they use terms, relative to some selected/current region. Examples of relative features: "Region with the highest value, besides the current region", "Value of the current region divided by the sum of values for all regions besides the current region".

2.2.4. Feature vector

Each data source contributes some elements to the feature vector. Simple data sources just provide one or more numerical values. For sources that provide multiple values (e.g. the language stats) the value ratios are also added. Regional data sources (which, for each website, can provide some value for any region) each contribute 15 values to the feature vector. For a given website, let $v(r)$ be the value that the data source has for some region r (e.g. the number of zip codes from r). Let $p(r)$ be the prior probability of a region r (region frequency in the training set). When applied to a set of regions R these functions are just summed: $v(R)=\Sigma_{r\in R}v(R)$, $p(R)=\Sigma_{r\in R}p(r)$. Used region sets: All is the set of all regions; *Cur* is the region of the current <site, region> pair; *Rest=All\Cur*; *Rival* is the non-current region with

the highest value: *Rival*=argmax *value*(*r*), *r*∈*All*; *RRest*=*Rest**Rival*. Let *ratio*(*R*)=*v*(*R*)/*v*(*All*), *rel*(*R*)=*v*(*R*)/*v*(*Cur*), *nratio*(*R*)=*ratio*(*R*)*p*(*All*)/*p*(*R*). When the 4 functions (*v*, *ratio*, *rel*, *nratio*) are applied to the 4 region sets (*Cur*, *Rival*, *Rest*, *RRest*) we get 16 values. But since *rel*(*Cur*) always equals 1, we exclude it.

We use the spectrums, produced out of the following data sources:

• **The number of mentions of the region's name in page titles; mentions of zip and phone codes in the website's address blocks.** We use publicly available official resources to build databases of zip and phone codes of different cities. Then we query the Yandex search engine to retrieve all webpages that have the name of a region close to the region's zip/phone code and the "address block markers" (e.g. "tel.», "apt.», "street», "st.», etc.). Searching for the region names requires a gazetteer with the names of the regions that need to be detected. The results for each website are aggregated by summing the corresponding values over the website's pages. The values are aggregated separately for all pages, for contact pages and for the index page. So, the contribution of this group of match values corresponds to 135 (3*3*15) features to the feature vector.

• **The number of mentions of the phone numbers registered in the region.** Phone numbers are extracted using a proprietary implementation of an entity extraction algorithm applied to all indexed web pages. The algorithm scans the texts of webpages to locate the phone-like sequences of numbers and symbols. These sequences are then compared to the list of phone schemes in order to select the sequences that are likely valid phone numbers and to extract the country code and area code information. This feature is a spectrum and thus contributes 15 values to the feature vector.

The following values are binary and also used to produce corresponding spectrums, as we previously described:

• **The match of the website's IP address.** IP addresses are being converted to regions using a custom IP-to-region database compiled from several sources. The primary source is the publicly available data from the RIPE Network Coordination Centre. The value shows if the website's IP address is from the range of IP address registered in the region under study.

• **The match of the website's *top level* domain.** Every website has Top Level Domain (TLD) as a part of its domain name. There are *country code TLDs* (ccTLDs) and *generic TLDs* (gTLDs). Some ccTLDs (e.g. .tv, .ws, .cc) are considered «vanity» and are widely used outside of the intended country, so they are not geo-coded. The value indicates whether the website's TLD is registered in the country under study.

• The regional result of a term-based classifier. We use a third-party proprietary implementation of a Bayesian text classification algorithm [13] that builds a classification model for every region present in the training data.

2.2.5. Non-spectrum features

We also use the following values as features, but do not produce spectrums (i.e. derived subset of features):

• **Site's language stats.** For each language that we can detect, we output the count and ratio of web site's pages in that language. The detector recognizes 45 languages, so this feature contributes 90 (45*2) values to the feature vector.

• **Results of a topical text-based classifier** [13] (briefly described previously). For each topic we output a single binary value. The classifier can recognize 632 topics and thus contributes 632 binary values to the feature vector.

• **Site's top level domain (TLD).** There is a link between the TLDs and countries beyond the ccTLD table (.mil and .gov are domains mainly used by US websites; .ru is often used by Ukrainian websites). Therefore we still use the raw TLDs as features. For each different TLD in our website collection we output a single binary value. The websites we want to classify have 194 different TLDs, so this feature contributes 194 binary values to the feature vector.

2.3. Training the model

To train the classification model we create a feature table with a feature vector for each <site, region> hypothesis. Not all <site, region> combinations are being considered though, as that would cause the unnecessary linear grow of the size and time requirements. Instead a group of *candidate regions* is formed for each website. We do not include a region unless there is any *hint* that the website could be associated with the region (e.g. some regional spectrum has a non-zero value associated with the region). That way, the average number of website's candidate regions is much lower than the total number of all possible regions. Moreover the increase in the number of possible regions does not impact the total number of candidate regions too much. If there are no hints, it's pointless to use the feature vector, generated for the hypothesis, to train or test the model as all the factor values would have the same empty state regardless of the region.

Our system uses the MatrixNet [11] machine learning algorithm that uses gradient boosting [5] to construct a strong learning model out of weak learners. Full binary decision trees [8] are used as the weak learners. By

applying the trained model to the feature vectors, we obtain a probabilistic score for each <site, region> pair. The resulting ranking of countries and cities w.r.t. the website is cut using selected thresholds for classification scores. The thresholds are selected to maximize the F1-score of the result on the training set.

2.3.1. Positive and negative examples

Each <site, region> pair created for websites from the training dataset is either correct or incorrect. Correct pairs are used as the positive examples for the machine learning algorithm while the incorrect are used as the negative examples. The number of negative examples is much higher (about 20 times) than the number of positive examples. To test the effect of the imbalance, we conducted an experiment where we significantly lowered the weights of the negative examples so that their combined weight matched the combined weight of the positive examples. The resulting optimal F_1-score remained the same, indicating that the result quality of our machine learning algorithm is not affected by this kind of class imbalance.

3. EXPERIMENT

3.1. Dataset

We used a high quality website directory internally maintained by the Yandex search engine. The directory is maintained and expanded by the professional editors and contains a collection of relevant web resources for the diverse set of topical and regional categories. Directory editors manually assigned a set of websites to regional and topical categories. The scopes used were *country*, *city* and *worldwide* (no country, no city). The experimental dataset contained all available regional data and consisted of about 115,000 websites linked to 344 regions of different scale (58 countries, 285 cities as well as the global and national categories). The cities corresponded to the countries where the Yandex operates e.g. Russia, Ukraine, Kazakhstan and Belarus. We used the publicly available official sources to build databases of zip codes, phone codes and TLDs as well as a region name gazetteer (in the official languages) for the regions of the dataset.

3.2. Experimental setup

3.2.1. One model per region type versus one model per region

As previously mentioned, we aimed to construct a classification model, which would be able to eliminate the problem of the lack of training data for less popular regions. So, our baseline system was identical to the pro-

posed one (same feature vectors and machine learning algorithm) except that it relied on an individual model for each region, not region type. The complexity of such baseline classification system is orders of magnitude higher than the complexity of our model-per-region-type system, giving it the advantage that it is able to focus on the specifics of each region. (For example, the complexity (the number of parameters) of each city's model in the "model per region" system was the same as the complexity of the single "cities" model in the "model per region type" system, so the overall complexity of our "cities" classification model was 285 times lower.)

3.2.2. Term-based features versus other features

Our system normally uses the output of an external term-based classifier as a feature. That classifier is built per-region, so one may argue that the system that uses its output does not adhere to the "one model per region type» idea. This is not the case since the term-based classifier is an opaque source external to the system. Nevertheless we still decided to evaluate the quality of our system without the term-based data features. We also measured the term-based classifier alone. Our aim was to show how combining the features in a single system allows us to achieve better classification quality.

3.2.3. Evaluation metrics

We used the F_1-score — a widely-used classifier performance evaluation measure (based on the precision and recall metrics) to evaluate the quality of our classification system and compare it to other systems. Using the F_1-score allows us to have a single performance characteristic that can be compared with other research results. Table 1 shows micro-averaged F_1-score values for each region type (countries, cities and the two special global classes — "global» and "national»). We also calculated macro-averaged F_1-score for the cities. We wanted to see how the size of the training set influences the classification quality. Figure 2 compares the F_1-scores of the systems for cities with different ranges of training set sizes. Figure 3 shows the margin of the performance increase that we gain with respect to the "one model per region" baseline performance for each individual city in our training set. We have chosen cities to demonstrate the advantage of our regional classifier. Countries typically have enough websites to build a good classifier using only websites with the regional focus on that country. At the same time, cities more often do not have enough training data and hence should benefit more from our approach (see Section 3.4).

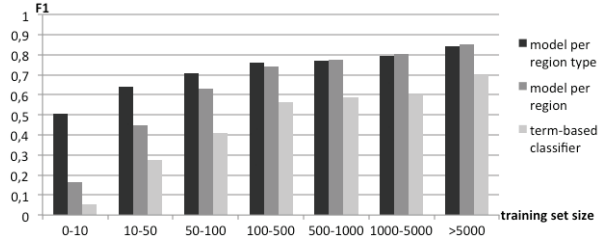

Figure 2. Relation between the training set size of regions and the performance of the systems under study

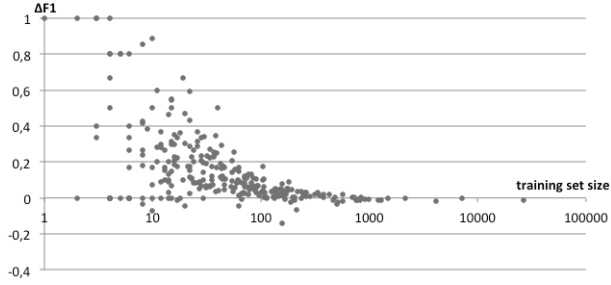

Figure 3. F_1-score gains over the baseline for different sizes of regions' training sets

Table 1. F_1-scores of different classification systems

	Countries	Cities		Global	National
		micro	macro		
Our system (Model per region type)	**0.93**	**0.83**	**0.68**	**0.65**	**0.70**
Our system w.o. term-based features	0.92	0.79	0.66	0.59	0.66
Model per region system	0.93	0.80	0.51	0.65	0.70
Term-based system	0.91	0.63	0.34	0.56	0.63

3.3. Classification without training

One interesting aspect of our classification system is the ability to detect regions that were not present in the train set. The classification model gets no direct information about the websites and regions that the feature vectors correspond to. The model just uses the features that reflect the relation

between the website and region. Thus, the model is insensitive to the actual regions that it assigns. We can use the model to check the relevance of any region, not just the regions that the model was trained against. To prove this, we conducted an experiment. We randomly partitioned the cities from the dataset into the train and test sets (maintaining the 2:1 ratio between the numbers of websites). We trained the city classification model using the websites from the training set and got the same performance on both set.

3.4. Results

The results show that our "model per region type» system produces better results than the "model per region» systems. The advantage becomes more apparent when we look at the classification quality for the regions that have different training set sizes. Both figures confirm our intuition that our system works much better than the baseline system for less popular regions. For the regions with large training sets the quality of our system is only marginally worse (the quality for countries and the worldwide/national categories is the same (see Table 1), while the quality for big cities drops no more than 0.015 F_1 score (see Figure 1 and Figure 2) w.r.t. the "model per region" system). Our system produces better results than the per-region systems for regions with *less than 500 training examples*. The advantage becomes more prominent as the sizes of regions' training sets continue to decrease. The quality of classification is rather high even when the region is almost or completely missing from the training set. While the micro-averaged F_1 score of the "model per region type" system is only 0.03 higher than that of the "model per region" system, the macro-averaged F_1 score increases significantly by 0.17.

4. CONCLUSION

We presented a regional classification system which uses classification models built per region type, rather than for each specific region. This allowed us to combine different types of features (entity- and term-based) in one classification framework and improve the classification performance for the websites from the regions that do not have enough training examples to build a region-specific classifier. The system was able to successfully detect regions that were not present in the training set.

In future, we plan to further improve the classification quality by incorporating websites' neighborhood analysis into the classification system.

REFERENCES

1. **E. Amitay, N. Har'El, R. Sivan, and A. Soffer.** Web-a-where: geotagging web content. SIGIR '2004.

2. **Z. Cheng, J. Caverlee, and K. Lee.** You are where you tweet: a content-based approach to geo-locating twitter users. CIKM '2010.
3. **D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg.** Mapping the world's photos. WWW '2009.
4. **J. Ding, L. Gravano, and N. Shivakumar.** Computing geographical scopes of web resources. In Proceedings of the 26th International Conference on Very Large Data Bases, VLDB '00.
5. **J. H. Friedman.** Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29, 2000.
6. **T.-Y. Liu.** Learning to rank for information retrieval. Found. Trends Inf. Retr., 3, 2009.
7. **X. Qi and B. D. Davison.** Web page classification: Features and algorithms. ACM Comput. Surv., 41, 2009.
8. **J. R. Quinlan.** Induction of decision trees. Mach. Learn., 1:81–106, March 1986.
9. **P. Serdyukov, V. Murdock, and R. van Zwol.** Placing flickr photos on a map. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '09.
10. **W. Zong, D. Wu, A. Sun, E.-P. Lim, and D. H.-L. Goh.** On assigning place names to geography related web pages. In Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries, JCDL '05.
11. **A. Gulin, P. Karpovich.** Greedy function optimization in learning to rank. Lection on the RuSSIR 2009 conference. http://romip.ru/russir2009/slides/yandex/lecture.pdf
12. **A. Pyalling, M. Maslov, P. Braslavski.** Automatic geotagging of Russian web sites. . In Proceedings of international conference on World Wide Web, WWW '06
13. **M. Maslov, A. Pyalling.** KC-classifier for web site classification. Russian Workshop on Information Retrieval Evaluation. ROMIP 2010